Матрици и Вектори

Реализация на класове, които позволяват операции с матрици и вектори от произволен тип. Типът трябва да има дефинирани аритметичните операции +,-,*,/ и <, ==, >

1. Клас Matrix : Представя матрица чрез двумерен масив

Голяма четворка:

```
Matrix();
Matrix(int,int);
Matrix(const Matrix &);
~Matrix();


Matrix& operator=(const Matrix &);
```

Селектори:

```
int getRows()const { return rows;}
int getCols()const {return cols;}
```

Оператор за прибавяне или изваждане на матрица ако 2те са с еднакви размерности:

```
void operator+=(const Matrix &);
Matrix operator+(const Matrix &);
void operator-=(const Matrix &);
Matrix operator-(const Matrix &);
```

Прибавяне или изваждане на число от всеки елемент на матрицата:

```
void operator+=(const T &);
Matrix operator+(const T &);
void operator-=(const T &);
Matrix operator-(const T &);
```

Умножение на матрица с матрица:

```
void operator*=(const Matrix &);
Matrix operator*(const Matrix &);
```

Умножение и деление на матрици с числа:

```
void operator*=(const T &);
Matrix operator*(const T &);
void operator/=(const T &);
Matrix operator/(const T &);
```

Достъп до елемент:

```
T* operator[](int) const ;
```

Транспониране:

```
void operator~();
```

Диагонализиране на матрица:

```
void diagonalize();
```

намиране на детерминанта:

```
double getDeterminant() const;
```

Намиране на обратна матрица:

```
Matrix operator!();
```

Извеждане към изходен поток:

```
friend ostream& operator<<(ostream & os, const Matrix<T> & m)
```

Въвеждане от входен поток:

```
friend istream& operator>>(istream & is, Matrix<T> & m)
```

2.Клас Vector : представя вектор чрез едномерен масив:

голяма четворка:

```
Vector();
Vector(int);
Vector(int,T*);
Vector(const Vector &);
~Vector();
Vector& operator=(const Vector &);
```

събиране и изваждане на вектори:

```
void operator+=(const Vector &);
Vector operator+(const Vector &);
void operator-=(const Vector &);
Vector operator-(const Vector &);
```

събиране, изваждане, умножение, и деление на вектор с число

```
void operator+=(const T &);
Vector operator+(const T &);
void operator-=(const T &);
Vector operator-(const T &);
void operator*=(const T &);
Vector operator*(const T &);
void operator/=(const T &);
Vector operator/(const T &);
```

достъп до елемент на вектора:

```
T& operator[](int);
```

скаларно и векторно произведение:

```
double operator*(const Vector &);
void operator^=(const Vector &);
Vector operator^(const Vector &);
```

нормализиране на вектор:

```
void operator!();
```

умножение на вектор с матрица:

```
void operator*=(const Matrix<T> &);
Vector operator*(const Matrix<T> &);
```

извеждане към изходен поток:

```
friend ostream& operator<<(ostream & os, const Vector<T> & v)
```

въвеждане от входен поток:

```
friend istream& operator>>(istream & is, Vector<T> & v)
```